

REMARKS

Prior to entry of this response, Claims 1-45 were pending in the application. No claims are added or cancelled herein. Hence, Claims 1-45 are still pending in the application upon entry of this response.

Claim 24 is amended herein, merely to correct a typographical error. Therefore, no new matter is introduced in the application by way of this claim amendment, and entry is kindly requested.

SUMMARY OF THE REJECTIONS/OBJECTIONS

Claims 1-13, 16-36, and 39-45 were rejected under 35 U.S.C. §103(a) as allegedly unpatentable over Chau et al. (“*Chau*”; U.S. Patent No. 6,721,727) in view of Jain (“*Jain*”; U.S. Patent Application Publication No. 2002/0073091) and further in view of Cheng et al. (“*Cheng*”; U.S. Patent No. 6,567,819); and

Claims 14, 15, 37 and 38 were rejected under 35 U.S.C. §103(a) as allegedly unpatentable over *Chau* in view of *Jain* and *Cheng* and further in view of Ng et al. (“*Ng*”; U.S. Patent No. 6,385,618) and further in view of Bowen et al. (“*Bowen*”; U.S. Patent No. 6,094,694).

Claim 24 was objected to because of informalities.

PRIORITY DATE

As a preliminary matter, Applicants traverse the priority date given the present application. As noted in the Office Action, the present application claims priority back to U.S. Provisional Appl. No. 60/204,196, filed May 12, 2000. Applicants acknowledge that, as stated in the Office Action, “the disclosure of the invention in the parent application and in the later-filed application must be sufficient to comply with the requirements of the first paragraph of 35

U.S.C. §112.” However, this does not mean that the disclosures of the parent and later-filed applications “must be the same” or that the later-filed application is not entitled to the priority date of a previous application if the disclosures of the later-filed application and the previous application to which the later-filed application claims priority are not the same.

As long as (a) the disclosure of the previous application, to which the later-filed application claims priority, is sufficient to comply with the requirements of the first paragraph of 35 U.S.C. §112, and (b) there is a valid chain of priority claims between the previous application and the later-filed application (e.g., from the continuing application to the parent non-provisional application, and from the parent non-provisional application to the provisional application), then the later-filed application deserves the priority date of the previous application. In the present application, both (a) and (b) are met and therefore, the present application is entitled to the earliest effective filing date (May 12, 2000) of U.S. Provisional Application No. 60/204,196. Hence, the earliest effective priority date for the present application is the date of the provisional application, May 12, 2000, not December 28, 2001.

THE REJECTIONS BASED ON THE PRIOR ART

Rejections under 35 U.S.C. §103(a)

(I) Claims 1-13, 16-36, and 39-45

Claims 1-13, 16-36, and 39-45 were rejected under 35 U.S.C. §103(a) as allegedly unpatentable over *Chau* in view of *Jain* and further in view of *Cheng*. Applicants traverse this rejection on the grounds that a prima facie case of obviousness is not established based on the cited references.

(A) Overview of General State of the Art and Various Embodiments of the Invention

It is well-known that with classic (i.e., conventional) object-oriented programming techniques, an instance of a class inherits attributes and methods from the class. However, as

stated in the Background section, with classic object oriented programming, inheritance cannot be used to provide different attributes for instances from the same class. Thus, because classic inheritance does not allow for instances of the same class to have different attributes, there is a need for per-instance attributes.

The application describes mechanisms for enabling per-instance methods and attributes. A mechanism for implementing per-instance attributes involves using “categories,” where an object inherits properties associated with a category class when the object is associated with a corresponding category object (page 8, lines 20-22). In other words, a category class provides a mechanism for categorizing an object (page 10, line 9). A purpose of the category mechanism is to offer another level of flexibility over and above what multiple inheritance offers. As such, the category mechanism allows for associating attributes with one instance of a “main” class without having to associate the same attributes with any other instances of the same “main” class.

This category mechanism is different than conventional class inheritance, wherein all objects of the same class inherit the same properties (i.e., attributes and methods) from the class and/or subclass from which the objects are instantiated. Thus, with conventional class inheritance, all objects of the same class are identical in their properties, with the exception of particular values for attributes and/or arguments for methods that may vary from object to object.

(B) How The Cited References Differ From the Claims

Claim 1 recites the following:

A computer-implemented method for establishing a structure of a data item within a computer system, where said data item is an instance of a first class and inherits attributes and methods from said first class, the method comprising the steps of:

creating a category object that is an instance of a category class, wherein said category class has one or more attributes; and
 associating said data item with said category object without associating said category object with all other instances of said first class thereby causing said data item to be associated with a structure that includes storage for values for said one or more attributes of said category class.

Claim 1 recites that the data item is an instance of a first class, but also is associated with a data structure for storage of values for one or more attributes of a different class, the category class. Furthermore, this data structure for storage of values for attributes of the category class is not associated with all instances of the first class. Thus, via use of the category class, attributes can be inherited by instances outside the instances' conventional class lineage inheritance mechanism. Stated otherwise, Claim 1 recites a method for establishing per-instance attributes. A practical result of establishing per-instance attributes is that one object of the first class can be associated with a category object and, therefore, inherit properties of the category class, whereas another object of the same first class is not associated with the category object and does not inherit the properties of the category class.

(i) The *Chau* reference

Generally, *Chau* describes an “XML system” that provides mechanisms for storing and retrieving XML documents in a relational database (i.e., DB2 XML Extender from IBM Corp.; *see*, e.g., col. 5, line 35-col. 6, line 13). The XML system is described as providing storage of XML data in a relational database as a non-traditional data type or as traditional data in relational tables. *Chau* may discuss that, in some cases, XML documents conform to corresponding Document Type Definitions (DTDs). Thus, one could suggest that an XML document is an instance of an DTD. It seems that the Office Action is contending that the “order.xml” document is an instance of the “LineItem.dtd” DTD.

However, as mentioned above, Claim 1 recites a programmatic mechanism for enabling per-instance attributes, through which an instance of a category class is associated with a particular instance of a “primary” class, but not all instances of the primary class. Hence, the particular instance can inherit attributes from the category class, in addition to the attributes and methods it inherits from its primary class. *Chau* does not disclose an instance of a class inheriting attributes from a class (i.e., a category class) that is not in the conventional lineage of the instance, nor does *Chau* disclose attributes from a class (i.e., the category class) that can be inherited by instances of different classes in different lineages. This category mechanism is different than conventional class inheritance, wherein all objects of the same class inherit the same properties (i.e., attributes and methods) from the class and/or subclass from which the objects are instantiated. Thus, with conventional class inheritance, all objects of the same class are identical in their properties, with the exception of particular values for attributes and/or arguments for methods that may vary from object to object.

For *Chau* to teach or fairly suggest per-instance attributes in a combination with *Jain* and *Cheng*, to the level required of the obviousness standard, it seems *Chau* would need to disclose that the “order.xml” document not only conforms to and ‘inherits’ attributes (and methods) from the “LineItem.dtd” DTD, but that “order.xml” also ‘inherits’ attributes from some DTD or schema other than “LineItem.dtd” (e.g., the category class, in addition to the first class of Claim 1). *Chau* simply does not disclose or suggest such a per-instance attribute mechanism, nor does any combination of *Chau* and *Jain* and *Cheng*. In fact, *Chau* teaches away from such a mechanism, in disclosing that (a) a “Valid Document” is “[a]n XML document that has an associated DTD [and that] [t]o be valid, the XML document cannot violate the syntactic rules specified in its DTD” and that (b) a “XML Attribute” is “[a]ny attribute specified by the ATTLIST under the XML element in the DTD” (col. 11, lines 7-10

and 13-14; emphasis added) and that an XML element “must be defined in the specified DTD” (col. 15, lines 24-25).

(ii) The *Jain* reference

Based on the priority claim discussed above, it is submitted that *Jain* does not qualify as prior art to the present application. However, even if Jain does qualify as prior art, it does not teach or fairly suggest the per-instance attribute mechanism of the embodiment recited in Claim 1.

Generally, *Jain* describes a translation tool for converting an XML DTD associated with an XML document to Java classes, from which a Java object corresponding to the XML document is instantiated (para. [0028]), and converting an XML document to a Java object (para. [0030]. Thus, *Jain* may provide a little better support than *Chau* for the position that an XML document is an object of a class and inherits attributes and methods from the class. However, *Jain* does not disclose an instance of a class inheriting attributes from a class (i.e., a category class) that is not in the conventional lineage of the instance, nor does *Jain* disclose attributes from a class (i.e., the category class) that can be inherited by instances of different classes in different lineages.

Jain seems to disclose that nodes of an XML document (Address, Street, City, State, Zip, and Country) are converted to respective corresponding Java classes (para. [0042]; block 516 of FIG. 5), that the classes for child XML nodes (Street, City, State, etc.) are used to generate class attribute variables for a parent XML node (Address) (para. [0056]; block 632 of FIG. 6) from which an object 124 is instantiated and from which the original XML document is accessed to set the data in the object 124 (para. [0044]-[0046] and [0056]; block 520 of FIG. 5). However, for *Jain* to teach or fairly suggest per-instance attributes in a combination with *Chau* and *Cheng*, to the level required of the obviousness standard, it seems *Jain* would at least need

to disclose that an XML document conforms to attributes specified in multiple DTDs or that one, but not all, corresponding Java object inherits attributes from multiple Java classes translated from corresponding XML DTDs. *Jain* does not disclose or suggest such a per-instance attribute mechanism, nor does any combination of *Chau* and *Jain* and *Cheng*.

For example, *Jain* does not teach or fairly suggest that one instance of the “Address” Java class (parent node) uses class attributes from respective Java classes “Street,” “City,” “State” and “Country” (child nodes) and that another instance of the “Address” Java class does not use class attributes from each respective child class. Rather, all instances of the “Address” Java class (parent node) would inherit the same attributes from the respective classes corresponding to the child nodes, basically because the classes corresponding to both parent and child nodes are derived from the same specific XML document structure. Stated otherwise, because *Jain* does not disclose a first XML document ‘inheriting’ attributes from a set of multiple DTDs and a second XML document ‘inheriting’ attributes from a subset of the multiple DTDs, and because the Java objects are instantiated from Java classes that are derived directly from a data structure specified in an XML DTD to which both the first and second XML documents conform, *Jain* does not disclose that a first Java object inherits attributes from a set of multiple Java classes and a second Java object of the same class inherits attributes from only a subset of the multiple Java classes. Rather, all objects of the same “Address” class inherit the same attributes, “Street,” “City,” etc. from the same single class, rather than a first object inheriting attributes from class “Address” and a second object inheriting attributes from class “Address” and some other class.

(iii) The *Cheng* reference

Generally, *Cheng* describes defining classes, including methods, using XML. For example, *Cheng* discusses defining class “ebObj.Purchase” from the XML statements in

TABLE I (col. 4, line 52-col. 5, line 22), where class “ebObj.Purchase” includes a method “closeDeal.” Additionally, *Cheng* describes defining an instance of a class (an object) using XML. Furthermore, *Cheng* cursorily mentions that additional method definition information pertinent only to a particular object may be added to an object definition (col. 6, lines 46-49). Thus, *Cheng* may provide support for per-instance methods. However, methods are clearly different from attributes because attributes, rather than methods, require allocation of storage for values. *Cheng* does not disclose a method for per-instance attributes, i.e., an instance of a class inheriting attributes from a class (i.e., a category class) that is not in the conventional lineage of the instance, or attributes from a class (i.e., the category class) that can be inherited by instances of different classes in different lineages. Hence, *Cheng* does not teach or fairly suggest causing a data item to be associated with a structure that includes storage for values for said one or more attributes of said category class, different from the data item’s “main” class from which it is instantiated.

For *Cheng* to teach or fairly suggest per-instance attributes in a combination with *Chau* and *Jain*, to the level required of the obviousness standard, it seems *Cheng* would at least need to disclose that an object definition (e.g., <obj type= “ebObj.Purchase”> of TABLE II) adds an object-specific attribute to a corresponding object (e.g., object AXN10009), where the object-specific attribute is not defined in the parent class from which the object is instantiated (e.g., the “ebObj.Purchase” class of TABLE I). *Jain* simply does not disclose or suggest such a per-instance attribute mechanism, nor does any combination of *Chau* and *Jain* and *Cheng*.

In view of the foregoing, a prima facie case of obviousness is not established for Claim 1 based on the *Chau*, *Jain*, and *Cheng* references, because no possible combination of these references teaches or fairly suggests all of the limitations recited in Claim 1. Claims 2-13 and

16-23 depend directly or indirectly from Claim 1 and, therefore, are patentable over the cited references of record for at least the same reasons as Claim 1. Claim 24 recites an embodiment for a per-instance attribute and, therefore, is patentable over the cited references of record for at least the same reasons as discussed herein in reference to Claim 1. Claims 25-36 and 39-45 depend directly or indirectly from Claim 24 and, therefore, are patentable over the cited references of record for at least the same reasons as Claim 24. Therefore, reconsideration and withdrawal of the rejection of Claims 1-13, 16-36, and 39-45 under 35 U.S.C. §103(a) is kindly requested.

Furthermore, each of Claims 2-13, 16-23, 25-36 and 39-45 includes at least one other limitation that makes it further patentable over the references of record. However, due to the fundamental difference between Claim 1 and the cited references discussed above, discussion of these additional differences is unnecessary and is foregone at this time. However, the rejection of the dependent claims is collectively traversed, and no statements of official notice or allegations of well-known features or overarching allegations of inherency or obviousness that may be present in the Office Action are stipulated to or admitted as prior art features, and the right to separately argue such features in the future is not disclaimed.

(II) Claims 14, 15, 37 and 38

Claims 14, 15, 37 and 38 were rejected under 35 U.S.C. §103(a) as allegedly unpatentable over *Chau* in view of *Jain* and *Cheng* and further in view of *Ng* and further in view of *Bowen*. Applicants traverse this rejection on the grounds that a prima facie case of obviousness is not established based on the cited references.

Claims 14 and 15 depend from Claim 1, and Claims 37 and 38 depend from Claim 24. Therefore, Claims 14, 15, 37 and 38 are patentable over the cited references of record for at least the same reasons as the respective independent claim from which this claim depends.

Furthermore, each of Claims 14, 15, 37 and 38 includes at least one other limitation that makes it further patentable over the references of record. However, due to the fundamental difference between Claim 1 and the cited references discussed above, discussion of these additional differences is unnecessary and is foregone at this time. However, the rejection of the dependent claims is collectively traversed, and no statements of official notice or allegations of well-known features or overarching allegations of inherency or obviousness that may be present in the Office Action are stipulated to or admitted as prior art features, and the right to separately argue such features in the future is not disclaimed. Reconsideration and withdrawal of the rejection of Claims 14, 15, 37 and 38 under 35 U.S.C. §103(a) is requested.

CONCLUSION

For the reasons set forth above, it is respectfully submitted that all of the pending claims (1-45) are in condition for allowance. Therefore, the issuance of a formal Notice of Allowance is believed next in order, and that action is most earnestly solicited.

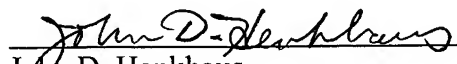
The Examiner is respectfully requested to contact the undersigned by telephone if it is believed that such contact would further the examination of the present application.

Please charge any shortages or credit any overages to Deposit Account No. 50-1302.

Respectfully submitted,

HICKMAN PALERMO TRUONG & BECKER LLP

Date: 4/17/06


John D. Henkhaus
Reg. No. 42,656

2055 Gateway Place, Suite 550
San Jose, CA 95110-1089
(408) 414-1080
Facsimile: (408) 414-1076

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Mail Stop Amendment, Commissioner for Patents, P. O. Box 1450, Alexandria, VA 22313-1450

on April 17, 2006

by


Daren Sakamoto